jc714 U.S. PTO

03/31/00

04-05-00

A

IN THE UNITED STATES PATENTAND TRADEMARK OFFICE
UTILITY PATENT APPLICATION TRANSMITTAL UNDER 37 CFR 1.53(b)

Address to:	*Attorney Docket No.*	2207/6856
Assistant Commissioner for Patents	*Inventor(s)*	ARIEL BERKOVITS
Box Patent Application	*Express Mail Label No.*	EL372085009US
Washington, DC 20231	*Total Pages*	23

Title of Application:

METHOD FOR REDUCING AN IMPORTANCE LEVEL OF A CACHE LINE

JC564 U.S. PTO 09/539839 03/31/00

Transmitted with the patent application are the following:

- __2__ Page(s) Transmittal form (and one copy)
- __16__ Page(s) Cover Page (1), Specification (10), claims (4), abstract (1)
- __3__ Page(s) Informal drawing
- __2__ Page(s) Declaration and Power of Attorney (unsigned)
- _____ Page(s) Recordation of Assignment and Assignment form
- _____ Page(s) Information Disclosure Statement (IDS) (copies of citations not included in number of pages)
- _____ Page(s) Certified copy of:

This application is a ___ Continuation / Continuation-in-Part / Divisional of prior application Serial No. _______, filed _____.

Fee calculation for large entity:	***No. Filed***	***No. Allowed***	***No. Extra***	***Rate***	***Fee***
Basic Fee					$690.00
Total Claims	30	20	10	× 18.00	$180.00
Independent Claims	5	3	2	× 78.00	$156.00
Multiple Dependent Claim				+ 260.00	
				Assignment	
				Total	$1,026.00

The Commissioner is hereby authorized to charge payment of any fees associated with this communication or arising during the pendency of this application per 37 CFR §1.16–1.21, or to credit any overpayments, to Deposit Account __11-0600__.

Express Mail Certificate

I hereby certify that the above paper/fee is being deposited with the United States Postal Service "Express Mail Post Office to Addressee" service under 37 CFR 1.10 on the date indicated below and is addressed to the Assistant Commissioner for Patents, Washington, DC 20231

Date of deposit: March 31, 2000

Person mailing paper/fee: Pilar Rodriguez

Signature Pilar Rodriguez

Respectfully submitted,

Shawn O'Dowd

Shawn W. O'Dowd (Reg. No. 34,687)
Attorney for Applicant(s)

Kenyon & Kenyon
333 W. San Carlos Street, Suite 600
San Jose, CA 95110
(408) 975-7500 - Telephone
(408) 975-7501 - Facsimile

SJ01 17345v1

UNITED STATES PATENT APPLICATION
FOR

METHOD FOR REDUCING AN IMPORTANCE LEVEL OF A CACHE LINE

INVENTOR:

ARIEL BERKOVITS

2207/6856
EL372085009US

Method for Reducing an Importance Level of a Cache Line

Field of the Invention

The present invention relates generally to a cache associated with a computer central processor unit (CPU), and, in particular, to a method for reducing the importance level of a cache line.

Related Technology

As is known, a cache is a fast local storage memory used in computer systems. The cache is typically arranged between the CPU and the main system memory. The cache is used to hold copies of data that are frequently requested from the main system memory by the CPU. A memory system can consist of several levels of caches. The lower the level of a cache, the closer that cache level is to the CPU and the faster and smaller the cache may be.

A common measure of cache performance is the "hit rate." When the CPU requests data from the main system memory, the cache control logic checks if the information is available in the cache memory. A cache hit occurs when the information requested by the CPU is in the cache. The cache responds to a hit by passing the requested information back to the CPU. The CPU receives the data relatively fast so it can handle it with a relatively short delay.

If the data requested by the CPU is not in the cache, a “miss" occurs. The data requested must then be retrieved from the slower main system memory or from a higher level of cache. A cache may be divided into a number of “lines,” or entries. A line of cache may hold data for more than one memory access. Typically, a copy of the retrieved data is saved into the cache memory in a cache line, overwriting the data currently existing in that line. Due to cost considerations, the cache memory is of limited size. Therefore, a so-called replacement policy, or algorithm, is used to determine which line of the cache memory is to be replaced when data is retrieved either from the main system memory or

from a higher level of cache.

The cache-hit rate is defined to be the percentage of memory requests that were completed by accessing the cache without going to higher cache level or to the main memory. High cache-hit rate results in higher overall CPU performance.

The replacement policy used by the cache has a direct effect on the hit rate of the cache. For example, replacing data that will be needed subsequently in a given program or process results in a lower hit rate since the CPU will then later not find the needed data in the cache memory. The CPU will have to go to the (slower) main system memory to retrieve the needed data. Thus the replacement policy affects hit rate and, consequently, overall CPU performance.

A variety of replacement policies are known. For example, the least recently used (LRU) policy replaces the cache entry which was less recently used compared to other cache entries. The LRU policy is based on the theory that the least recently the data was used, the less likely the program will request it again. Another replacement policy is the random policy, which selects cache memory locations for replacement at random.

The replacement policy implemented in a given cache is typically fixed in the cache hardware. The application programmer writing software to run on the CPU associated with the cache has no way to provide an indication to the cache that a given line of cache is a good candidate for replacement independent of the particular replacement policy in effect.

Summary of the Invention

The present invention provides a method for reducing an importance level of a line in a memory of a cache, the method comprising providing an instruction to the cache indicating that the line is a candidate for replacement.

Brief Description of the Drawings

Fig. 1 shows a schematic diagram of a cache associated with a main system memory and a CPU according to an embodiment of the present invention;

Fig. 2 shows a table demonstrating prior art cache line replacement for a cache set and memory access sequence for an LRU replacement policy; and

Fig. 3 shows a table demonstrating cache line replacement when a method for reducing an importance level of a cache line according to an embodiment of the present invention is applied to the cache set, memory access sequence, and LRU replacement policy of Fig. 2.

Detailed Description

Referring to Fig. 1, cache 14 is connected to CPU 12 via bus 18 and to main system memory 16 via bus 20. Instruction storage medium 6 is read by input/output device 8, which feeds instructions stored on input/output device 8 into CPU 12. Instruction storage medium 6 may be any type of suitable medium for storing instructions of a program, such as, for example, a magnetic storage disk. Input/output device 8 may be any type of suitable device such as, for example, a disk drive. CPU 12 may be any type of appropriate CPU, such as a processor or microprocessor. Main system memory 16 may be any type of appropriate storage medium, such as dynamic random access memory (DRAM), for example. Cache 14 includes cache control logic 24 and cache memory 26. Cache 14 may be any type of appropriate cache system. Cache memory 26 may be static random access memory (SRAM), for example. As embodied herein, cache memory 26 is part of a first cache level. Other, higher levels of cache memory may be provided.

An instruction according to an embodiment of the present invention, hereinafter referred to as the reduced importance cache line (RICL) instruction, may be an independent memory access instruction. Alternatively, the RICL may be a part of, or an extension of, another memory access instruction, such as, for example a 'store' instruction. The RICL is decoded by the decoder of CPU 12 and sent to a memory control unit (MCU) associated

with the CPU with an address which is the parameter of the instruction. The MCU then executes the instruction.

As embodied herein, each location in main system memory 16 can map to only a subset of the total number of cache entries, or lines. Each of these subsets is collectively known as a "set." Control bits associated with a cache set indicate which entry of the set will be allocated for this memory data, replacing a copy of data already in that cache line. As embodied herein, a fixed heuristic function is used as a replacement policy to set the value of the control bits according to the history of memory requests. There is, as is typical, no way to directly control those bits using software.

Reference may now be had to Figs. 2 and 3 to demonstrate how an RICL instruction according to an embodiment of the present invention may be used to decrease the number of memory requests from CPU 12 completed by accessing cache 14 without going to a higher cache level or to main system memory 16, and thereby increase cache hit rate.

Fig. 2 shows a table demonstrating a prior art cache line replacement for a cache set and memory access sequence using an LRU replacement policy. A sequence of eleven memory accesses { a, b, a, c, d, b, b, e, a, c, d} are mapped to the same four-line cache set {0, 1, 2, 3}. Each of { a, b, a, c, d, b, b, e, a, c, d} indicate a main memory location being accessed by the CPU. It is assumed that the cache set {0, 1, 2, 3} initially contains copies of data for locations w, x, y and z, respectively, i.e., cache line 0 corresponds to memory location w, cache line 1 corresponds to memory location x, cache line 2 corresponds to memory location y and cache line 3 corresponds to memory location z.

Columns 30-41 in Fig. 2 represent:

• in row P, the sequence of eleven memory accesses, sequentially from left to right;

• in row Q, the allocation of the memory access retrievals when the memory access required access to the main system memory, i.e., in which cache line of cache set {0, 1, 2, 3} the retrieved data is saved;

• in rows R, S, T, U, the ranking of the cache lines of cache set {0, 1, 2, 3} based on the control bits according to the LRU replacement policy, row R indicating the least recently used cache line, row S indicating the next least recently used cache line and row U indicating the "most" recently used cache line of the set, i.e., least recently used increasing from bottom to top; and

• in row V, the main memory location for which data was replaced under the least recently used replacement policy.

Initially, cache set {0, 1, 2, 3} contains copies of data for locations w, x, y and z, respectively, and the LRU replacement policy ranking is cache lines 0, 1, 2, 3 (see column 30). Upon the first memory access, for main system memory location a (row P, column 31), the data for location w in cache line 0 is replaced with a copy of the data from main system memory location a, since cache line 0 is the least recently used cache line, as indicated by the 0 in row R, column 30. The replacement of data for location w is indicated by the w in row V, column 31. According the LRU replacement policy, cache line 1 then becomes the least recently used cache line, as indicated by the 1 (column 31) taking the place of 0 in row R. Similarly, upon the second memory access, for main system memory location b (row P, column 32), cache line 1 is replaced with a copy of the data from main system memory location b, since cache line 1 is the least recently used cache line, as indicated, as noted above, by the 1 in row R of column 31. The data for location x is thereby replaced, as indicated in row V, column 32.

Upon the third memory access, for main system memory location a (row P, column 33), the data for location a is already present in cache line 0, so no access of the main system memory, and hence no replacement of a cache line, is necessary.

In the complete access sequence depicted in Fig. 2, it is apparent from row V that a total of eight cache entry replacements are necessary (w, x, z, y, a, b, c, d).

Referring now to Fig. 3, a table similar to that shown in Fig. 2 is presented. Fig. 3 depicts the same memory access sequence, with the same LRU policy, as that shown in Fig. 2. In this case, however, an RICL instruction according to an embodiment of the

present invention is implemented together with the seventh memory access (row P, column 37). The RICL instruction here has the effect of moving the cache line (1) containing a copy of the data for main memory location b to the top of the LRU ranking (row R, column 37). Thus, in the eighth memory access (row P, column 38), the data for b in cache line 1 is replaced (see row V, column 38) instead of the data for a in cache line 0, as with the "pure" LRU replacement policy, as shown in Fig. 2 (see row V, column 38 of Fig. 2).

The RICL instruction might be used as shown in Fig. 3 because the data for main system memory location b will not be used as soon as other data, such as location a, by an application running on the CPU. As a result of location b, rather than location a, data being replaced (see row V, column 38 of Figs. 2 and 3), fewer total cache line replacements, i.e., cache misses, occur. Implementation of the RICL instruction according to an embodiment of the present invention has the advantageous affect in this example of reducing the number of cache entry replacements from eight to five. The result is a higher hit rate and, consequently, improved performance of CPU 12.

An RICL instruction according to an embodiment of the present invention may advantageously be implemented in an application kernel running on CPU 12. For example, CPU performance for a matrix multiplication function could be improved using the RICL instruction. Shown below are two code sequence loops for a matrix multiplication C = A x B, where each line of A is multiplied by all line of B to form the first line of C, then next line of A is multiplied by all lines of B to form the second line of C, etc. Code Sequence I is a basic matrix multiplication loop, while Code Sequence II is the same matrix multiplication loop with use of the RICL instruction.

Code Sequence I

```
For (int i = 0; i < SIZE; i++){
     For (int j = 0; j < SIZE; j++){
          For (int k = 0; k < SIZE; k++){
               // C[i][j] += A[i][k]*B[k][j];
               Load r1 ← A[i][k];
               Load r2 ← B[k][j];
```

```
                R3 ← r1 * r2;
                Load r4 ← C[i][j];
                R3 ← r3 + r4;
                Store C[I][j] ← r3
            }
        }
    }
```

Code Sequence II

```
    For (int i = 0; i < SIZE; i++){
        For (int j = 0; j < SIZE-1; j++){
            For (int k = 0; k < SIZE; k++){
                // C[i][j] += A[i][k]*B[k][j];
                Load r1 ← A[i][k];
                Load r2 ← B[k][j];
                R3 ← r1 * r2;
                Load r4 ← C[i][j];
                R3 ← r3 + r4;
                Store.RICL C[I][j] ← r3
            }
        }
        // Assume: j = SIZE - 1
        For (int k = 0; k < SIZE; k++){
            // C[i][j] += A[i][k]*B[k][j];
            Load.RICL r1 ← A[i][k];
            Load r2 ← B[k][j];
            R3 ← r1 * r2;
            Load r4 ← C[i][j];
            R3 ← r3 + r4;
            Store.RICL C[I][j] ← r3
        }
    }
```

In Code Sequence II, the RICL instruction, or indication, is asserted for every A line the last time it is used. Lowering the importance of used A and C cells, frees space for more B cells in the cache, decreasing the number of main system memory accesses and thereby increasing the cache hit rate.

Thus, an instruction according to the present invention provides information to the

cache about an unneeded cache line. A parameter of the instruction is a memory address. The cache associates the memory address with a cache line if it exists in the cache. The instruction indicates that the memory address will not be used in the near future. Therefore, the importance of the cache line, if any, holding this memory address can be reduced. The information provided by the instruction does not affect the semantics of an application program being run on the CPU associated with the cache, but will provide a useful hint to the cache so as to increase hit rate and, thereby, CPU performance. The instruction will not cause exceptions in the CPU operations.

Execution of the instruction may result in a change in the cache control bits that track memory requests from the CPU so as to optimize the allocation of cache lines. As noted above, a memory access may be smaller than the size of a cache line. The cache control logic may reduce the importance of a cache line based on the first indication to any byte of a cache line, after indication to the entire cache line, or after any number of the bytes in the cache line are indicated to be less important. Alternatively, the cache control logic may ignore an indication provided by the instruction entirely. Additionally, the indication provided by the instruction can propagate to higher levels of cache.

An instruction according to the present invention may be advantageously used in application kernels. As is known, application kernel is a small portion of software that consumes a large number of cycles of the CPU in a typical usage of the application. Because kernels are typically hand written in assembler language, the developer has the knowledge about the application and the ability to schedule instructions, such as an RICL instruction according to the present invention. An RICL instruction according to the present invention could also be applied in compilers, especially feedback driven compilers, or other interpreter of a higher-level language.

An instruction according to the present invention may reside on any suitable instruction storage medium, such as, for example, a magnetic storage disk, as would be understood by one of skill in the art.

Variations may be made in specific implementations that are within the scope of the

present invention. For example, a method according to the present invention may be an addition of a hint bit to an existing memory access instruction. The bit indicates that this access is the "last" access, for now, to this memory location and the corresponding cache entry is a good candidate for replacement. It should also be emphasized that, although an LRU replacement policy was described herein, a method according to the present invention may be applied with any suitable replacement policy and/or cache allocation methodology. An instruction according to the present invention provides an indication that a cache line is a candidate for replacement. The cache control logic may use the instruction to alter the cache allocation methodology in other ways besides mere replacement of a cache line, as would be understood by those of skill in the art.

WHAT IS CLAIMED IS :

1. A method for reducing an importance level of a line in a memory of a cache, the method comprising providing an instruction to the cache indicating that the line is a candidate for replacement.

2. The method as recited in claim 1 further comprising reducing an importance level of the line based on the instruction.

3. The method as recited in claim 2 wherein the reducing of the importance level of the line results in the line being replaced prior to another line scheduled for replacement by a replacement policy of the cache.

4. The method as recited in claim 3 wherein the replacement policy is a least recently used policy and wherein the other line is less recently used than the line.

5. The method as recited in claim 1 further comprising altering an allocation methodology of the cache based on the instruction.

6. The method as recited in claim 1 wherein the instruction in part of an application kernel.

7. The method as recited in claim 1 wherein the instruction is generated by a compiler.

8. The method as recited in claim 1 wherein the instruction is an extension of a memory access instruction.

9. A instruction for increasing hit rate of a cache, the instruction comprising an indication that a line in a memory of the cache is a candidate for replacement.

10. The instruction as recited in claim 9 wherein the indication causes a reduction of an importance level of the line.

11. The instruction as recited in claim 10 wherein the reducing of the importance level of the line results in the line being replaced prior to another line scheduled for replacement by a replacement policy of the cache.

12. The instruction as recited in claim 11 wherein the replacement policy is a least recently used policy and wherein the other line is less recently used than the line.

13. The instruction as recited in claim 9 further comprising altering an allocation methodology of the cache based on the instruction.

14. The instruction as recited in claim 9 wherein the instruction in part of an application kernel.

15. The instruction as recited in claim 9 wherein the instruction is generated by a compiler.

16. The instruction as recited in claim 9 wherein the instruction is an extension of a memory access instruction.

17. An article comprising a storage medium, the storage medium having a set of instructions, the set of instructions being capable of being executed by at least one processor to implement a method for reducing an importance level of a line in a memory of

a cache, the set of instructions when executed comprising providing an indication to the cache that the line is a candidate for replacement.

18. The article as recited in claim 17 wherein the set of instructions further comprises reducing an importance level of the line based on the indication.

19. The article as recited in claim 18 wherein the reducing of the importance level of the line results in the line being replaced prior to another line scheduled for replacement by a replacement policy of the cache.

20. The article as recited in claim 19 wherein the replacement policy is a least recently used policy and wherein the other line is less recently used than the line.

21. The article as recited in claim 17 further comprising altering an allocation methodology of the cache based on the indication.

22. The article as recited in claim 17 wherein the indication is part of an application kernel.

23. The article as recited in claim 17 wherein the indication is generated by a compiler.

24. The article as recited in claim 17 wherein the indication is an extension of a memory access instruction.

25. A cache comprising:

a cache memory including a cache line; and

a cache control logic for reducing an importance level of the cache line based on an instruction.

26. The cache as recited in claim 25 wherein the instruction provides an indication that the cache line is a candidate for replacement.

27. The cache as recited in claim 26 wherein the cache control logic reduces an importance level of the cache line based on the indication.

28. The cache as recited in claim 27 wherein the reducing of the importance level of the cache line results in the cache line being replaced prior to another cache line scheduled for replacement by a replacement policy of the cache.

29. The cache as recited in claim 25 further comprising altering an allocation methodology of the cache based on the instruction.

30. A method for reducing an importance level of a line in a memory of a cache, the method comprising:

providing an instruction to the cache indicating that the line is a candidate for replacement; and

reducing an importance level of the line based on the instruction.

Abstract

A method for reducing an importance level of a line in a memory of a cache. An instruction is provided to the cache, the instruction indicating that the line is a candidate for replacement. The importance level of the line may then be reduced based on the instruction. The method may increase cache hit rate and, hence, microprocessor performance.

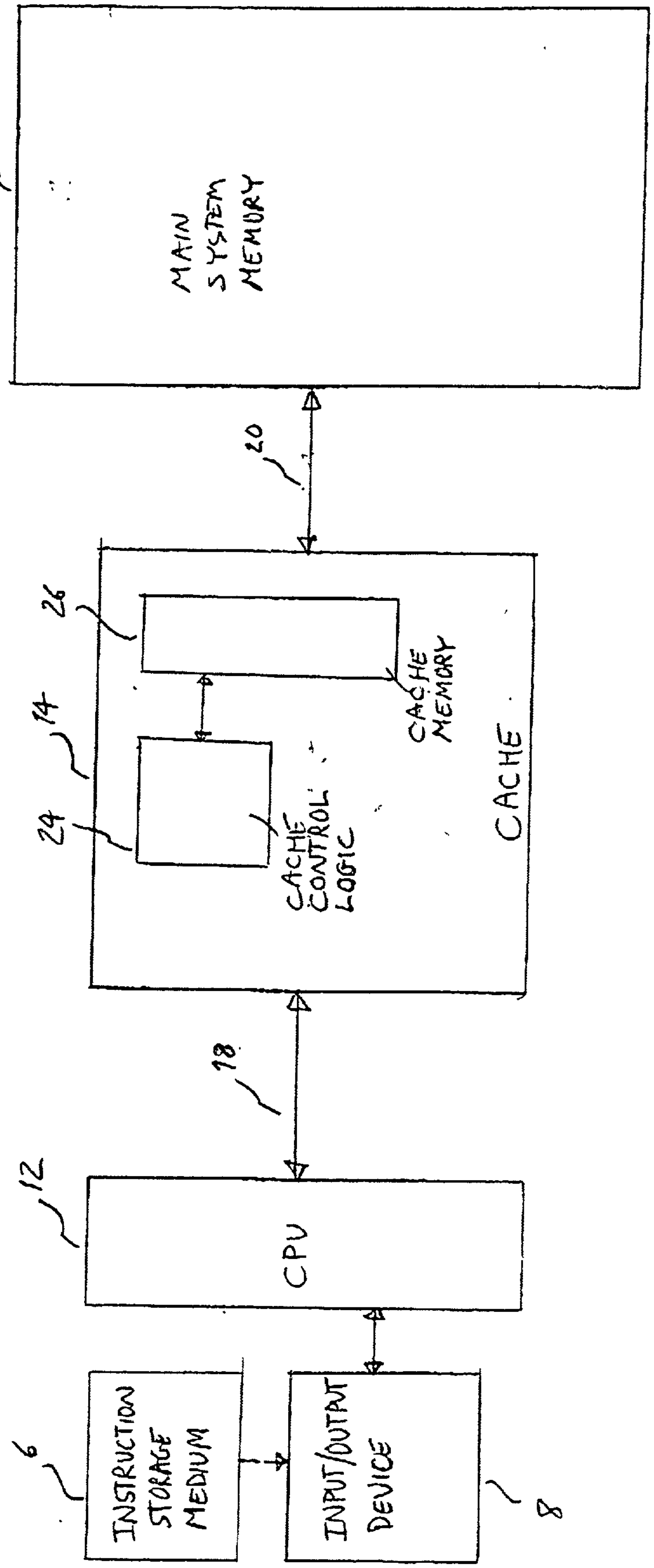
16
MAIN SYSTEM MEMORY
20
26
CACHE MEMORY
14
24
CACHE CONTROL LOGIC
CACHE
18
12
CPU
6
INSTRUCTION STORAGE MEDIUM
INPUT/OUTPUT DEVICE
8

FIG. 1

	30	31	32	33	34	35	36	37	38	39	40	41
P	Sequence	a	b	a	c	d	b	b	e	a	c	d
Q	Allocation	a=>0	b=>1		c=>2	d=>3			e=>0	a=>2	c=>3	d=>1
LRU => R	0	1	2	2	3	1	0	0	2	3	1	0
S	1	2	3	3	1	0	2	2	3	1	0	2
T	2	3	0	1	0	2	3	3	1	0	2	3
U	3	0	1	0	2	3	1	1	0	2	3	1
V	Replaced	w==>	x==>		y==>	z==>			a==>	c==>	d==>	b==>

PRIOR ART

Fig. 2

2207/6856
EL372085009US

37

38

P	Sequence	a	b	a	c	d	b	RICL(b)	e	a	c	d
Q	Allocation	a==>0	b==>1		c==>2	d==>3			e==>1			
LRU => R	0	1	2	2	3	1	0	1	0	2	3	1
S	1	2	3	3	1	0	2	0	2	3	1	0
T	2	3	0	1	0	2	3	2	3	1	0	2
U	3	0	1	0	2	3	1	3	1	0	2	3
V	Replaced	w==>	x==>		y==>	z==>			b==>			

Fig. 3

PATENT
Docket No. 2207/6856

DECLARATION AND POWER OF ATTORNEY FOR PATENT APPLICATION

As a below named inventor, I hereby declare that:

My residence, post office address and citizenship are as stated below next to my name.

I believe I am the original, first and sole inventor (if only one name is listed below) or an original, first and joint inventor (if plural names are listed below) of the subject matter which is claimed and for which a patent is sought on the invention entitled:

METHOD FOR REDUCING AN IMPORTANCE LEVEL OF A CACHE LINE

the specification of which is attached hereto unless the following is entered:

was filed on	as United States Application Number or PCT International Application Number	and was amended on (if applicable)

I hereby state that I have reviewed and understand the contents of the above identified specification, including the claims, as amended by any amendment referred to above.

I acknowledge the duty to disclose information which is material to patentability as defined in 37 CFR §1.56.

PRIOR FOREIGN APPLICATION(S)

I hereby claim foreign priority benefits under 35 USC §119(a-d) or §365(b) of any foreign application(s) for patent or inventor's certificate, or §365(a) of any PCT International application which designated at least one country other than the United States, listed below and have also identified below any foreign application(s) for patent or inventor's certificate, or PCT International application having a filing date before that of the application on which priority is claimed:

Application Number	Country	Filing Date (day/month/year)	Priority Not Claimed

PROVISIONAL APPLICATION(S)

I hereby claim the benefit under 35 USC §119(e) of any United States provisional application(s) listed below:

Application Number	Filing Date

PRIOR UNITED STATES APPLICATION(S)

I hereby claim the benefit under 35 USC §120 of any United States application(s), or §365(c) of any PCT International application designating the United States, listed below and, insofar as the subject matter of each of the claims of this application is not disclosed in the prior United States or PCT International application in the manner provided by the first paragraph of 35 USC §112, I acknowledge the duty to disclose information which is material to patentability as defined in 37 CFR §1.56 which became available between the filing date of the prior application and the national or PCT International filing date of this application:

Application Number	Filing Date	Status (patented, pending, abandoned)

POWER OF ATTORNEY

I hereby appoint the following attorney(s) and/or agent(s) to prosecute this application and to transact all business in the Patent and Trademark Office connected therewith:

Paul H. Heller (Reg. No. 21,074); John C. Altmiller (Reg. No. 25,951); Shawn W. O'Dowd (Reg. No. 34,687); David E. Foster (Reg. No. 41,759); Jerray Wei (Reg. No. 43,247) of KENYON & KENYON with offices located at 1500 "K" Street NW, Suite 700, Washington, DC, 20005-1257, telephone (202) 220-4200, and at 333 W. San Carlos Street, Suite 600, San Jose, CA, 95110-2711, telephone (408) 975-7500;

and James E. Jacobson, Jr. (Reg. No. 31,626); Thomas C. Reynolds (Reg. No. 32,488); Raymond J. Werner (Reg. No. 34,752); Richard C. Calderwood (Reg. No. 35,468); Joseph R. Bond (Reg. No. 36,458); Naomi Obinata (Reg. No. 39,320) of INTEL CORPORATION.

PATENT
Docket No. 2207/6856

DECLARATION AND POWER OF ATTORNEY FOR PATENT APPLICATION (Cont.)

Direct telephone calls to:	Send correspondence to:
SHAWN W. O'DOWD (408) 975-7500	KENYON & KENYON 333 W. San Carlos, Street, Suite 600 San Jose, CA 95110-2711

I hereby declare that all statements made herein of my own knowledge are true and all statements made on information and belief are believed to be true; and further that these statements were made with the knowledge that willful false statements and the like so made are punishable by fine or imprisonment, or both, under §1001 of Title 18 of the United States Code and that such willful statements may jeopardize the validity of the application or any patent issuing thereon.

Full name of first or sole inventor	Last Name **BERKOVITS**	First Name **ARIEL**	Middle Name
Residence	City **YUVALIM**	State or Country **ISRAEL**	Country of Citizenship **ISRAEL**
Post Office Address	Street **YUVALIM 240**	City **YUVALIM**	State or Country & Zip Code **ISRAEL 20142**
Signature		Date	
Full name of second inventor	Last Name	First Name	Middle Name
Residence	City	State or Country	Country of Citizenship
Post Office Address	Street	City	State or Country & Zip Code
Signature		Date	
Full name of third inventor	Last Name	First Name	Middle Name
Residence	City	State or Country	Country of Citizenship
Post Office Address	Street	City	State or Country & Zip Code
Signature		Date	
Full name of fourth inventor	Last Name	First Name	Middle Name
Residence	City	State or Country	Country of Citizenship
Post Office Address	Street	City	State or Country & Zip Code
Signature		Date	